Appears in the 9th IEEE International Symposium on High Performance Distributed Computing.

An Evaluation of Alternative Designs for a Grid Information Service

Warren Smith[†], Abdul Waheed[*], David Meyers[‡], Jerry Yan

[†]Computer Sciences Corporation
[*]MRJ Technology Solutions
[‡]Directory Research L.L.C. [1]

NASA Ames Research Center
Moffett Field, CA 94035-1000

{wwsmith,waheed}@nas.nasa.gov {dmeyers,jyan}@arc.nasa.gov

Abstract

Computational grids consisting of large and diverse sets of distributed resources have recently been adopted by organizations such as NASA and the NSF. One key component of a computational grid is an information services that provides information about resources, services, and applications to users and their tools. This information is required to use a computational grid and therefore should be available in a timely and reliable manner. In this work, we describe the Globus information service, describe how this service is used, analyze its current performance, and perform trace-driven simulations to evaluate alternative implementations of this grid information service. We find that the majority of the transactions with the information service are changes to the data maintained by the service. We also find that of the three servers we evaluate, one of the commercial products provides the best performance for our workload and that the response time of the information service was not improved during the single experiment we performed with data distributed across two servers.

1. Introduction

Computational grids [3] consisting of large and diverse sets of distributed resources have recently been adopted by organizations such as NASA in their Information Power Grid (IPG) [7,9] and NSF in their Partnership for Advanced Computing Infrastructure (PACI) effort [11,12]. The key middleware supporting computational grids is the Globus toolkit [4]. The Globus toolkit provides services such as security, communication, managing distributed applications, remote data transfer, and information. The increase in the number of resources and users in Globus-based computational grids has highlighted deficiencies in the current implementation of some of these services. In particular, the implementation of the Globus Grid Information Service (GIS) was insufficient to handle the loads being placed on it until the recent addition of a second server. The result of the high load on the GIS was that queries made by users were not being fulfilled in a timely manner and therefore, users could not effectively locate and determine how to access the resources available in the Globus-based computational grids.

The goal of this study is to examine the demands made on the Globus GIS and evaluate how well different GIS implementations can meet those demands. We begin in Section 2 by describing the current Globus GIS and how it is used by Globus software and users. In Section 3, we use trace data obtained from the Globus GIS to study the load that is placed on the GIS and characterize who is accessing the GIS for what reasons. We find that the majority of the accesses made to the GIS are for the purposes of modifying the information stored in the GIS. This is contrary to the assumption made by most of the commercial software used to implement grid information services which assume that the vast majority of the operations will be searches. This access pattern is also similar to the access patterns expected for Directory Enabled Networking [8] and our results should therefore apply to that domain. We also find that fairly high demands are placed on the GIS: there are typically 90 connections open to the GIS and 8.8 operations per second occur on average.

Section 4 describes how we use trace-driven simulation to evaluate GIS configurations and presents the results of these simulations. We find that of the three servers we evaluate, the server provided by Vendor 1 exhibits the best performance on the hardware we used for evaluation. We also find that the communication

[1] This work was partially funded by grant 08.008.005.002 from the Research Institute for Advanced Computer Science at NASA Ames Research Center.

latency between the clients and the servers has a significant impact on performance and that distributing the GIS data across two servers increases the average update time by 27 percent and decreases the average search time by 76 percent. Section 4 describes our methodology for evaluating GIS implementations, presents the results we have obtained, and analyzes these results. Section 5 describes the changes to Globus in version 1.1.3 that will affect the Globus GIS and discusses the effects we believe they will have on GIS performance and reliability. Section 6 presents our conclusions and future work.

2. Metacomputing Directory Service

The Metacomputing Directory Service (MDS) [2] is the grid information service of the Globus project. The MDS is a repository of information for using computational grids. It contains information about organizations, people, computers, networks, software, applications, and project-specific data. The MDS is accessed using the Lightweight Directory Access Protocol (LDAP) [5,6] and data in the MDS is organized as entries in a hierarchical tree called the directory information tree. The location of an entry in the directory information tree is based on organizations and other entries it is associated with. For example, a Portable Batch System scheduler interface for a SGI Origin computer system at NASA Ames would be located in the directory information tree (moving towards the root of the tree) at: service=jobmanager-pbs, hn=origin.arc.nasa.gov, ou=Ames Research Center, o=National Aeronautical and Space Administration, o=Globus, c=US. Each entry in the tree is a set of attributes where an attribute has a name and one or more values. The names are text strings and the values can be of any number of pre-defined types, but are typically strings. For example, the entry for the above PBS scheduler interface might contain the name of the host it is running on, the port it is listening on, the type of scheduler available, how many nodes are managed and available through the interface, properties of the scheduler, and so forth. Details of the MDS directory information tree and the types of entries that are defined are available in [2].

The LDAP protocol supports addition, deletion, and modification of entries in the directory information tree and allows clients to search an LDAP server for entries that satisfy specified search constraints. The LDAP communications between the client and the server can be unauthenticated, authenticated with an identity and password, or authenticated with an identity and password over a Secure Sockets Layer (SSL) connection. Currently, the MDS is not using SSL connections. LDAP databases and clients are provided by many vendors. In this work, we are concerned with implementations provided by OpenLDAP and two companies, Vendor 1 and Vendor 2, that we cannot identify for licensing reasons. Currently, the Globus software uses the OpenLDAP client software to access the MDS, which is contained in two Netscape LDAP servers located at the National Center for Supercomputing Applications (NCSA) in Champaign-Urbana, Illinois.

As the number of entities participating in the Globus computational grid increased, the response time of a single server became inadequate. Fortunately, LDAP servers support several techniques to improve response times. First, data can be distributed across several computer systems. This is accomplished by placing sub-trees of the directory information tree on different hosts and accessing these sub-trees through LDAP referrals. For example, all data from NASA Ames can reside in a database on a LDAP server running on a machine at NASA Ames. This approach may increase the amount of adds, deletes, and modifies that can occur in a given time interval but may increase the response time of searches if the searches have to contact multiple servers to get their results.

Second, Data can be replicated on multiple hosts. This approach may increase the number of searches that can be serviced in a given time by servicing the searches on different hosts, but it may decrease the number of adds, deletes, and modifies that can occur in a given time since any changes made must be propagated to the replications. Third, the LDAP servers can be tuned to improve their response time. This tuning varies from implementation to implementation, but one example is the creation of indexes that allow for faster searches. As mentioned above, the Globus MDS data has recently been distributed across two servers at NCSA and this has significantly improved the access times to the data contained in the MDS.

There are several different Globus software modules that update information in the MDS. First, when a computer system is initially added to the Globus grid, a setup script populates the MDS with information about the host, it's network interfaces, the networks it is attached to, and the Globus software running on the host. Second, there are a set of scripts that are run periodically to update information about the computers, networks, and software available through the grid. Third, there are MDS updates associated with the Globus Resource Allocation Manager (GRAM) [1]. The GRAM is used to start applications on remote computer systems and there are two GRAM components on remote computer systems that interest us here. The GRAM job manager is a daemon that is started for each application. The job manager starts, monitors, and manages an application and informs a second software module, the GRAM reporter, of the application state. Periodically, the GRAM reporter will determine the number of available nodes on the computer

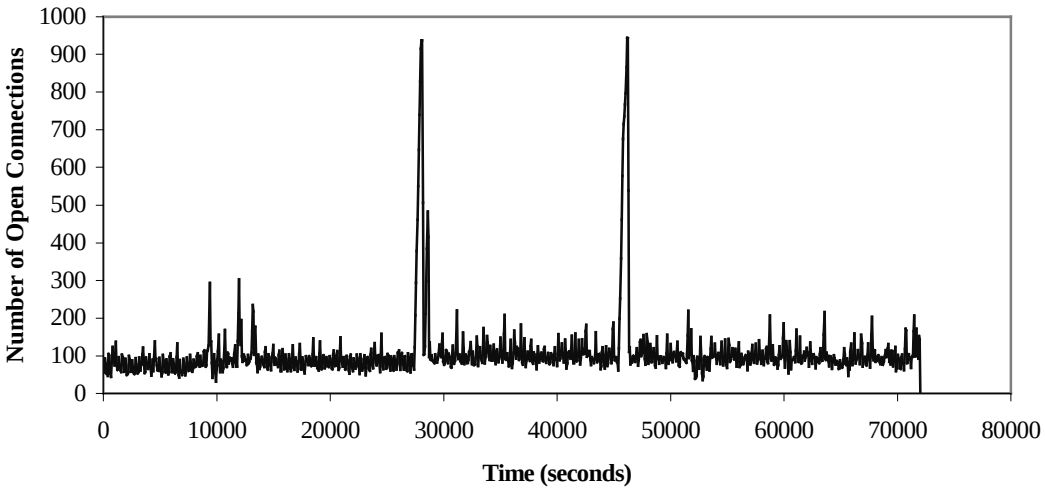

Figure 1. The number of open connections to the MDS server over time.

system it is associated with, determine the status of any queues associated with the GRAM, determine the users that can use the GRAM, gather the state of the applications submitted through the GRAM, and update all of this information in the MDS. By default, this process is performed every 30 seconds and user and job information is not published into the MDS. Many sites do not publish the user information for security reasons, and do not publish job information due to the load it places on the MDS.

There are many possible ways that users can use the MDS data. One common way is that when a user uses the globusrun program to start an application on a host, the user can specify the hostname, and globusrun will contact the MDS to find the host name, port number, and other information necessary for GRAM to start an application on the remote computer. Another common use of the MDS is to query the status of applications that are started on remote systems. Users have not typically employed the MDS in more sophisticated ways because the response time of the MDS was not sufficient to support these activities.

A new release of Globus, version 1.1.3, should occur in June or July of 2000 and there are changes that affect the MDS in this release. The most significant change is that the MDS will be a highly distributed information service with an OpenLDAP server running on every host that provides compute cycles to remote users through Globus. These LDAP servers will maintain local data and can also be configured to push data to organizational LDAP servers that maintain data from a group of hosts that are running the Globus software. A further description of this approach is provided in Section 5 along with our thoughts on this approach.

3. Workload Characterization

In this section, we analyze 20 hours of trace data recorded from the Globus MDS server. This data consists of all of the accesses to the LDAP server from when the server restarted on February 24, 2000 to when it restarted again on February 25. These 20 hours of data contain 86,695 connections to the server and 143,446 adds, deletes, modifies, and searches. If we also consider connects, binds to an identity, responses to requests, unbinds from an identity, and closes then there are 633,672 operations in the workload or an average of 8.8 operations per second.

Figure 1 graphs the number of open connections at any given time. The data shows that there are typically 90 connections open at any time with two spikes of over 900 active connections. Figure 2 presents a histogram of connection duration. This data shows that there are a large number of relatively short-duration connections. In fact, 88 percent of the connections last less than 120 seconds and 97 percent of the connections last less than 240 seconds. Examining the data closer, we determine that the long-duration connections are those where a user connects to the information service and periodically searches for the state of their application using that connection. Figure 3 shows a histogram of the number of adds, deletes, modifies, or searches per connection. The data shows that the vast majority of the connections have relatively few operations. In fact, 97 percent of the connections have two or less of these operations. Examination of this data

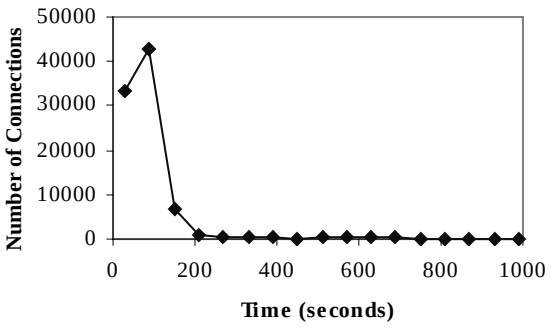

Figure 2. Histogram of connection durations.

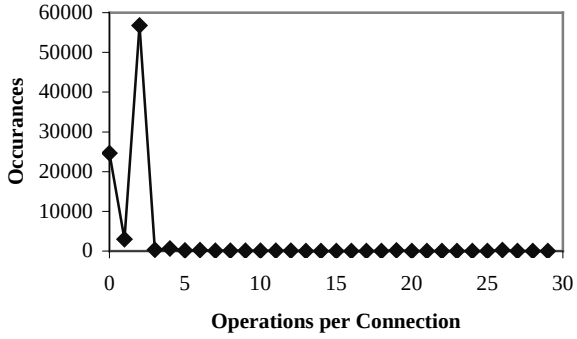

Figure 3. Histogram of operations per connection.

shows that the connections with the most number of operations are those where a user is periodically searching for the state of their application. There are also a fairly high number of operations per connection when a gram reporter updates information when publishing of job information is enabled.

Each connection consists of a connect, a bind to an identity, one or more adds, deletes, modifies, or searches, an unbind, and a close. Table 1 shows the number of add, delete, modify, and search operations in the trace data. As one can see, the majority of the operations are modifies. These modifications come from GRAM reporters to update information such as job status, the load on workstations, the nodes available through schedulers, and so forth. Modifications are also used to touch objects so that clients will know that Globus daemons were up in the recent past. There are relatively few entries added and deleted because only entries for jobs are added and deleted and very few computer systems are publishing this information due to the load it places on the MDS. Entries can also be added to the MDS when new organizations start using the MDS, but these events are relatively rare and did not occur in the trace data analyzed here. There are relatively few searches because at the time this data was recorded, users were avoiding searches of the MDS because these searches were not returning results for long periods of time. Table 1 also presents the number of errors that occur during the operations. The data shows that a high percentage of the add and search applications result in errors. Most of the errors that occur during add operations occur when Globus software first tries to modify an entry in the MDS, the modify fails, an add is attempted, and it also fails. The modify typically fails because the bind to an identity failed. These successions of failures can be avoided by responding correctly to the LDAP error codes that are generated: an add should only be attempted after a failed modify if the modify failed because the entry does not exist. The search operations also result in a high percentage of errors. Almost all of these errors are caused by the searches timing out before they complete because the server was too highly loaded.

We also use this trace data to classify the connections and identify what type of entity initiated the connection and for what purpose. These classifications are shown in Table 2. As one can see, we can classify almost 100 percent of the connections and the majority of the connections, 67 percent, are modifications of data made by GRAM reporters.

4. Experimental Analysis

We use a set of experiments to evaluate the performance of LDAP server implementations, implementation-specific LDAP configurations, and distribution of an LDAP directory information tree across multiple hosts. Our approach in this work is to evaluate GIS configurations by starting one or more LDAP servers that will act as the GIS on one or more systems, loading these servers with the contents of the Globus MDS as of February 24, 2000, and then replaying 20 hours of access that were made to the Globus MDS server between February 24, 2000 and February 25, 2000 from one or more workstations. The clients on the workstation that exercise the LDAP servers are written in Java and use the Java Naming and Directory Information Interface (JNDI). The trace data used for these simulations is the derived from the data that we analyzed in Section 3.

The data used in the simulation differs from the recorded data in that the recorded data does not include the actual modifications made to entries or the actual contents of the entries that were added to the LDAP servers. We construct this data off-line using the data in the MDS and our knowledge of which attributes Globus modifies. We perform different experiments and adjust the load on the LDAP servers by simulating the trace data faster or slower than real time. Our testing environment

Table 1. Occurrences of LDAP operations.

Operation	Number of Operations	Percent of Total Operations	Number of LDAP Errors	Percent of Operations Resulting in Error
Add	1044	0.73	943	90.33
Delete	81	0.06	6	7.41
Modify	134611	93.84	3807	2.83
Search	7710	5.37	5867	76.10
Total	143446	100.00	10623	7.41

Table 2. Classification of MDS connections.

Number of connections	Percent of Connections	Description
58476	67.45	Modification to job manager, the queues it can submit to, and the jobs it has submitted.
364	0.42	Modification of host information. This includes entries for the host, it's views from various networks, the Globus software running on the host.
73	0.08	Modification of network information.
7	0.01	Modification of software information.
71	0.08	Deletion of jobs.
191	0.22	Search for job status
50	0.06	Search for job managers.
1494	1.72	Search by the MDS monitor.
1234	1.42	Search for all GlobusPhysicalResource objects. We do not currently know what entity is generating these searches.
22	0.03	Search for all objects. These are most likely users performing tests.
0	0.00	Unclassified adds.
5	0.01	Unclassified deletes. Deletions are so infrequent that we do not classify them.
4440	5.12	Connections with a bind and unbind but no operations. We do not currently have an explanation for these connections.
20232	23.34	Connections containing a connect, a bind failure, an unbind, and a close.
86659	99.96	Classified connections.
86695	100.00	Total number of connections

consists of a Sun UltraSparc 30 with one 296 MHz CPU, 512 MB of memory and UltraSCSI disk drives running Solaris 2.6 where we run most of our experiments and a Sun UltraSparc 10 with a 333 MHz CPU, 128 MB of memory and a UltraSCSI disk drive that we use for experiments when two servers are used to provide an information service. We use several client systems in different geographical locations to test OpenLDAP server version 1.2.10 and the latest LDAP products from Vendor 1 and Vendor 2 running on one or both of these systems. We evaluate a MDS configuration using the response time of the LDAP commands and if the LDAP servers continue to operate.

There are many possible GIS designs and we only aim to evaluate a few in this work. First, we evaluate the relative performance of the OpenLDAP, Vendor 1, and Vendor 2 LDAP servers. Second, we evaluate the performance effects of using indexes to improve search performance. Third, we evaluate the performance of a GIS that distributes data over two LDAP servers using referrals. Fourth, we discuss the advantages and disadvantages of data replication in our environment.

4.1. Comparison of LDAP Servers

To compare the performance of the LDAP servers from OpenLDAP, Vendor 1, and Vendor 2, we start one of these servers on our test system, load the LDAP server with the MDS contents from February 24 as described above, and then use a simulator running on a workstation to exercise the LDAP server under test. The simulator replays the trace data recorded from the MDS server in real time or at a faster or slower rate. The simulator can

Table 3. Performance of operations on individual LDAP servers under varying loads.

Load	LDAP Server	Add (ms)	Delete (ms)	Modify (ms)	Search (ms)	Weighted Average (ms)
~0.5	OpenLDAP	101	108	317	1657	387
1.0	Vendor 1	121	106	159	2463	283
	Vendor 2	933	1230	903	1358	928
1.33	Vendor 1	194	141	233	2874	375
2.0	Vendor 1	270	351	581	4031	764
	Vendor 2	31722	36920	26558	22945	26407

also be programmed with constraints such as the maximum number of open connections at a time and the maximum number of new connections that can be opened a second. Table 3 summarizes the results of these experiments. The load column indicates the load that was placed on the server under test. For the Vendor 1 and Vendor 2 results, the load is the factor by which the simulator sped up time during the simulations. For the OpenLDAP server we indicate a load of about 0.5 but this does not directly relate to the amount of time it took to complete the simulation. We found that the OpenLDAP server failed when we attempted to perform simulations at a load of 1.0 or 0.5. The server failed by not responding to queries in the middle of the simulations. To complete a simulation, we limited the maximum number of open connections to 50 and the maximum number of new connections a second to 20, roughly half the average of 90 open connections that occur during a real-time simulation. These are the results reported for OpenLDAP in Table 3 and leads to our first result: the OpenLDAP server is the only server of the three we tested to fail under the loads we placed on it.

The data for the servers from Vendor 1 and Vendor 2 for a load of 1 indicate that the server from Vendor 1 performs adds, deletes, and modifies in 0.13 seconds on average, 9 times faster than the 1.2 seconds of the server from Vendor 2. The server from Vendor 2 performs searches in 1.4 seconds on average, which is 1.8 times faster than the 2.5 seconds of the Vendor 1 server. This data seems to indicate that the Vendor 2 server has been highly optimized for search performance. We note that the server from Vendor 2 has more indexes to improve search performance than the server from Vendor 1 (indexes and their performance effects are discussed further in Section 4.2) and changes to the data require that these indexes be updated. Optimizing search performance is an excellent characteristic for the typical data in LDAP servers that is not modified very often, but trading improved search performance for decreased modification performance is not the best choice in our environment. This observation is further emphasized when the response times for the simulations that execute in half of the time they were recorded (load of 2) is examined. The data shows that the response time from the server from Vendor 2 is 34.6 times slower than the server from Vendor 1. We observe that during the experiment with the server from Vendor 2, the CPU load on the host running the server had a load of about 5.5 (as measured by the Unix uptime command) while we observed a CPU load of at most 0.1 during the experiment with the server from Vendor 1. This data may indicate that the Vendor 1 server is I/O bound while the Vendor 2 server is CPU bound. Future experiment to further examine these observations would be to examine the performance of these servers on a multiprocessor system and/or a system with an array of disks.

4.2. Indexing

One technique that is used to improve the performance of searches is indexing. An index essentially stores search results for quick lookups when a search occurs. For example, an index can be maintained for an operating system attribute so that a search for all Solaris computer systems is quickly responded to by accessing the index. The index would be an equality index on the operating system attribute that would contain a list of entries associated with the value Solaris. These entries would be all of the entries in the directory that have a value of Solaris for the operating system attribute.

The disadvantage to indexes is that they have to be updated whenever an attribute they are indexing is changed. This adds overhead to the add, modify, and delete operations to maintain any indexes that refer to any of the attributes in the entries that are changed. We evaluate the performance of indexes by adding an index to the server from Vendor 1 to improve the performance of the 191 searches made to determine the status of jobs and then performing a real-time simulation. These searches are performed over the whole directory tree to look for entries with GlobalJobIDs that contain the name of the system that the job is executing on. To improve the performance of these searches, we add an approximate index on the GlobalJobID attribute.

We find that the search performance improves 79 percent from 2463 ms to 515 ms while the performance of the add, delete, and modify operations decreases 38 percent from 159 ms to 220 ms. We therefore see that

there is a significant improvement in search performance when an index is used, but there is a decrease in performance for changes to entries. In our environment, this decrease in performance for the large number of modifications outweighs the increase in performance of the search operations.

4.3. Data Distribution

Distribution of data can be used to support very large databases and to improve the performance of databases. Distributing data across multiple servers results in more resources being available to handle data modifications and hopefully better performance. Data distribution can also improve search performance when the searches access data from only a few servers. In this situation, more resources are available to satisfy searches. If searches access data from many servers, many transactions have to occur to obtain the search results and this reduces search performance.

We are interested in distributing data across multiple servers because of the large percentage of operations in our environment that change the data. Globus users observed a dramatic performance improvement when the Globus GIS recently moved from a single server to two servers. We wish to perform simulations to characterize the effects of distributing data across more than one server. We distribute our data across two servers from Vendor 1 in the same way that the Globus GIS currently distributes it's data: one server contains all of the data from NASA and the NSF Alliance sites, the second server contains all of the other data.

We performed a simulation in half of the time the data was recorded in. We assume that the components updating data in the information service will know which of the servers contains the data they are updating so that only one server will be contacted for each update. We find that distributing our data across two servers results in an increased update time of 27 percent. This surprising result may be due to a load imbalance on the two servers or due to the differing performance of the two computer systems the LDAP servers ran on. We are continuing to investigate this result. We also find that the average search time is 970 ms which is 76 percent faster than the 4031 ms search time when a single LDAP server is used.

Another way to use multiple computer systems to store our data is to replicate data on one or more servers. Replication improves search performance by having more resources available to perform searches and improves reliability by having data still be available when a server goes down. The disadvantage to replication is that when data is changed, these changes must be propagated to the replicas of the data and this adds overhead. At this time, we do not evaluate replication because of the relatively few number of searches in our workload and the relatively large number of modifications.

5. Globus 1.1.3 Grid Information Service

The Globus group has made several changes in Globus version 1.1.3 to attempt to improve the performance of their information service. The major change is that the default information service is highly distributed to lower the number of changes made to the data on any single server and eliminate the bottleneck caused by having many data updates go to only a few servers. By default, each host that supports application execution via Globus has a Grid Resource Information Server (GRIS) on it. The GRIS consists of the OpenLDAP front end layered over the GRAM reporter (described in Section 2) that provides information about the host the GRIS is running on including the host itself, the software on the host, the users who can access to the host, and the applications running.

The other new component of the Globus 1.1.3 GIS is organizational LDAP servers. An organizational sever is a LDAP server (Globus will configure an OpenLDAP organizational server if it is asked to) that contains referrals to the GRIS servers it is associated with. For example, an organizational server would contain an entry for each of the GRIS servers in that organization and each of these entries would refer to a GRIS server on a machine in the organization. This configuration results in a "pull" model for retrieving data: when a user performs a search, an organizational server queries the GRIS servers that may contain the data the user is interested in and then passes this data to the user. This is very different from the "push" model used by earlier Globus releases where the GRAM reporter pushed data to remote LDAP servers.

We have not evaluated these changes to the Globus information service using experiments but we do have some initial thoughts. First, having a large number of LDAP servers will mean fewer accesses to each server and therefore faster response times. The difficulty is that the data of interest to users is now widely distributed. If a user is interested in information from a small set of hosts, we do not believe it will add a large overhead to perform a small number of queries to different servers to find the information. If a user is interested in information that comes from a large number of hosts, we believe it will take an unacceptably long time to query all of the hosts that have the information. This is where organizational servers that aggregate grid information can improve search performance. Searches that examine data from a large number of hosts can query a smaller number of organizational servers to find their results in an acceptable period of time if the organizational servers cache the data they pull from GRIS servers. The next problem is that if users perform searches for dynamic information from

many hosts, this data cached in the organizational servers will not be up to date and must be pulled from the GRIS servers. This means that a search for dynamic information sent to an organizational server can require many pulls of data from GRIS servers and the potential benefits of having a LDAP server on each Globus host have been negated. If this situation occurs in practice, it implies that there is no reason to have an LDAP server on each Globus host.

To summarize our analysis, the effectiveness of the changes to the Globus information service in version 1.1.3 will depend on how users want to access data from this service. If users do not perform many searches for dynamic data that are produced by a significant number of hosts, then this approach should provide good performance. If users do wish to search for dynamic data produced by a significant number of hosts, the OpenLDAP servers on the Globus hosts will not improve performance and a set of organization servers that maintain up-to-date information should be used.

6. Conclusions

In this paper, we described our investigation of alternative designs for a grid information service. We described the Globus grid information service and how the Globus toolkit and users access this information service. We analyzed trace data obtained from the Globus information service and found that the majority of the operations are modifications of existing data, that the information service has roughly 90 connections open at any given time, and the information service is performing 8.8 operations per second. We described our methodology for experimentally evaluating LDAP server designs using trace data and contents obtained from the Globus grid information service and we evaluated the OpenLDAP server and two servers from vendors we cannot specify. We found that the OpenLDAP server failed when we attempted to place our recorded load upon it and that the server from Vendor 1 has 9 times lower response times than the server from Vendor 2 when modifying data in the directory service but the server from Vendor 2 has 1.8 times better search performance. If we double the load on these servers, the server from Vendor 1 has 35 times lower response times than the server from Vendor 2. We also observe that the Vendor 2 implementation seems to be highly optimized for searching and for executing on multiprocessor computer systems. We find that indexing can be used to reduce the response time of searches but adding index also results in slower response times when changes are made to the data because the indexes have to be kept up to date. Finally, we distributed our directory information tree across two servers on two computer systems and performed a simulation with twice the load of our trace data. We found that distributing data increases the response time of updates by 27 percent but decreases the response time for searches by 76 percent.

In future work, we will continue to evaluate different configurations for grid information services and we will investigate other factors that impact the performance of a grid information service such as an increase in the number of users and the use of secure connections the servers. To assist in this work, we plan to develop a system of synthetic grid entities to apply loads to proposed grid information service. The current Globus components use the grid information service in a relatively predictable way. This makes it relatively easy to develop synthetic components and have these components apply loads to the grid information service. Further, as described in our workload analysis, users also use the MDS in predictable ways. This allows us to develop synthetic users and evaluate the performance and fault tolerance of the design of a grid information system if there are hundreds or thousands of users. We expect the number of users of computational grids to greatly increase as the middleware grows in stability and more users observe the advantages of using computational grids.

References

[1] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, S. Tuecke. A Resource Management Architecture for Metasystems. *Lecture Notes on Computer Science*, 1998.

[2] S. Fitzgerald, I. Foster, C. Kesselman, G. von Laszewski, W. Smith, S. Tuecke. A Directory Service for Configuring High-Performance Distributed Computations. In *Proceedings of the 6th IEEE Symposium on High-Performance Distributed Computing*, pp. 365-375, 1997.

[3] I. Foster and C. Kesselman, editors. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kauffman, 1999.

[4] I. Foster and C. Kesselman. Globus: A Metacomputing Infrastructure Toolkit. *International Journal of Supercomputing Applications*. 11(2):115-128, 1997.

[5] T. Howes, M. Smith, G. Good. *Understanding and Deploying LDAP Directory Services*. Macmillan Technical Publishing, 1999.

[6] T. Howes and M. Smith. *LDAP: Programming Directory-Enabled Applications with Lightweight Directory Access Protocol*. Macmillan Technical Publishing, 1997.

[7] W. Johnston, D. Gannon, B. Nitzberg. Grids as Production Computing Environments: The Engineering Aspects of NASA's Information Power Grid. In *Proceedings of the Eighth IEEE International Symposium on High Performance Distributed Computing*, 1999.

[8] J. Strassner and F. Baker. *Directory Enabled Networks*. Macmillan Technical Publishing, 1999.

[9] The Information Power Grid. http://ipg.nasa.gov.

[10] The Globus Project. http://www.globus.org.

[11] The National Computational Science Alliance. http://www.ncsa.uiuc.edu/alliance.

[12] The National Partnership for Advanced Computing Infrastructure. http://www.npaci.edu.